Journal of Engg. Research **Vol. 4 No. (4) December 2016** pp. 24-45

السرية الأمامية والخلفية لشبكات الاستشعار اللاسلكية بطريقة إداه المفاتيح

إبراهيم الراشد، فاروق باغجي وإيمان القريشي
قسم هندسة الحاسوب، جامعة الكويت، الكويت

الخلاصة

شبكات الاستشعار اللاسلكية غير المراقبة هي نوع مختلف من شبكات الاستشعار اللاسلكية حيث تعمل أجهزة الاستشعار في بيئات معادية دون بالوعة ثابتة لتوجيه بيانات الاستشعار لها. فبدلا من ذلك تقوم أجهزة الاستشعار بتخزين البيانات المستشعرة الخاصة بها لحين مرور البالوعة النقالة وتحميل هذه البيانات. هذا الوضع يمكن أن يؤدي إلى خرق خصوصية البيانات، في حال اختراق جهاز الاستشعار، يمكن منها الوصول إلى البيانات المخزنة وأي مفاتيح تشفير يحتويها جهاز الاستشعار. لذلك، فمن الضروري القيام بضمان سرية البيانات التي تم إنشاؤها قبل اختراق جهاز الاستشعار، وتسمى بالسرية الأمامية، وكذلك سرية البيانات التي تم إنشاؤها بعد اختراقه، وتسمى بالسرية الخلفية. في هذا البحث، نقدم نظام متوزع يستخدم تطور مفتاح الشفرة لضمان السرية الأمامية، ويستخدم آلية تعاونية لتوزيع البيانات المستشعرة من أجهزة الاستشعار لضمان السرية الخلفية.

A key management approach for forward and backward secrecy in unattended WSNs

Ebrahim A. Alrashed*, Faruk Bagci and Eman Alquraishi

Computer Engineering Department, Kuwait University, Kuwait

**Corresponding author: ibrahim.esmael@ku.edu.kw*

ABSTRACT

Unattended wireless sensor networks are a different type of wireless sensor network, where sensors operate in hostile environments without a fixed sink to route the sensed data to. Alternatively, sensors accumulate and store their sensed data until a mobile sink visits the node and off load them. This situation can lead to breach of data privacy, when, upon a node compromise, an adversary can access the stored data and any encryption keys the node possesses. Therefore, it is essential to ensure the secrecy of data, which was generated before capture, termed forward secrecy, as well as secrecy of data generated after the node has been compromised, termed backward secrecy. In this work, we present a distributed scheme that utilizes key evolution to ensure forward secrecy, and uses a co-operative data distribution mechanism to ensure backward secrecy of the sensor node's data.

Keywords: Data confidentiality; data secrecy; key evolution; mobile sink; unattended wireless sensor networks.

INTRODUCTION

A wireless sensor network (WSN) (Yick *et al.*, 2008) is a wireless network consisting of a large number of small, battery-powered, low-cost, and spatially distributed sensor nodes that have the ability to sense their surrounding area for specific events, and to store and process the sensed data. Sensor nodes are used to monitor physical and environmental conditions such as temperature, sound, pollutants, pressure, vibration, image, or motion. Due to size and cost constraints, sensor nodes are typically resource limited and have limited computational power, memory and communication capacities. Typically in a WSN, a large number of sensor nodes are deployed in an area, where every sensor node can only communicate with other sensors within its coverage area (i.e., transmission range). The wireless sensor network is connected to one or more base stations that are usually referred to as *sinks*. The role of a sink is to collect the data from the sensor in the network and transmit the collected data to another base station to be processed, typically via an Internet, Satellite or GSM connection.

The unattended aspect of the network gives the adversary an additional advantage, particularly during the absence of sink. The adversary can move about the network and compromise sensors. In such environments, sensor nodes are highly susceptible to physical attacks or capture (Di Pietro *et al.*, 2008; Sen, 2009). In an unattended wireless sensor network (UWSN), sensor nodes are typically deployed in remote physical regions and are left unattended with little or no maintenance after installation, and are unable to deliver their sensed data in real-time. They operate in an untrusted and hostile environment, where the adversary can act maliciously and unobstructed. Unattended WSNs do not use a static sink, but rather use a mobile sink that periodically roam about the UWSN to collect sensed data from the sensor nodes which is stored locally or at some designated nodes within a network. Since the sink is not always available in the service area, the sensor nodes, cannot offload their sensed data to the sink in real-time. Consequently, each sensor node must store the sensed data locally in its memory for a considerable period of time until the next visit from the mobile sink. Intervals between successive sink visits present vulnerable periods for attacks, during which a mobile adversary can take advantage of the absence of the sink, compromise sensors and read all the stored information and leave before the sink visits the network and offloads the sensed data. Therefore, protecting the accumulated data in the sensor node's memory until the next sink visit is a critical issue in UWSNs.

Sensor nodes in WSNs' susceptibility to various kinds of attacks is due to several factors. Primarily, wireless communication between nodes makes it possible for other wireless devices to tune in and eavesdrop on network communications. The inherent resource limitations of sensor nodes make it difficult to implement complex security algorithms. Also, nodes are deployed in remote regions with little or no maintenance, making it easy for an adversary to capture, damage or tamper with a sensor node. In addition, an adversary might capture a node, learn its secrets, reprogram it and release it back into the network. In such cases of node compromise, it is essential to protect the confidentiality of the network data at all times. With respect to a compromised sensor, data can be classified into: before compromise, during compromise, and after compromise. During compromise, the adversary has full control and hence, data cannot be protected. However, data generated before and after compromise can be protected. Protecting the confidentiality of data generated before compromise is called *forward secrecy*, while protecting the confidentiality of data generated after compromise is called *backward secrecy*. In other words, forward secrecy deals with ensuring that pre-compromise data cannot be exposed, if a sensor is compromised and backward secrecy deals with ensuring that post-compromise data cannot also be revealed.

The contribution of this paper is that, it proposes a distributed protocol, which achieves forward and backward secrecy in UWSN under attack by malicious and compromising adversaries. We primarily base our proposed protocol on the following

concepts: key evolution, distribution of data, and sensor co-operation. This proposed protocol, guarantees sensed data security and reliability through a data distribution scheme, which stores the sensed data of one sensor node into another randomly selected node. This distribution of sensed data is carried out through secure communication based on evolving encryption keys.

The rest of the paper is organized as follows. In Section 2, we briefly review related work in the area of data confidentiality in UWSNs. The network model and threat model are presented in Section 3. Section 4 presents the proposed scheme. In Section 5, a security analysis of the proposed scheme is presented and Section 6, presents the experimental results. Finally, Section 7 concludes the paper.

RELATED WORK

Several techniques have been proposed in the literature for providing data secrecy in UWSNs (Di Pietro *et al.,* 2012; Di Pietro *et al.,* 2008(b); Ma & Tsudik, 2008; Ren *et al.*, 2010; Ren *et al.*, 2009). Di Pietro *et al.*, (2012) in their work, propose cooperative self-healing techniques to provide forward and backward secrecy. In these techniques, each sensor can be in one of three states: red, yellow or green. A red sensor is a currently compromised sensor, a yellow sensor is a previously compromised and released sensor whose current key is known by the adversary, and a green sensor is a not currently compromised sensor whose key is not known by the adversary. The idea of this technique is to update the key of the yellow sensors and heal those using contributions from the green sensors. Sensor nodes send random numbers that are used by the yellow sensors to compute the new key.

Two approaches have been proposed for node healing in UWSN. The nodes can explicitly request contributions from other sensors using a PUSH approach (Di Pietro *et al.,* 2008(b)), or sensors voluntarily send their random contributions to their neighbors using a PULL approach (Ma & Tsudik 2008). In both of these schemes, each time the sink arrives, it refreshes all the security information of the sensors. However, these schemes can only provide backward secrecy, if the adversary cannot eavesdrop on the communication between green and yellow sensors, where they have assumed that the adversary is not a global eavesdropper namely, it can only listen to the traffic sent to the currently compromised nodes. The authors have also proposed a public key scheme to overcome unreliable sensors problem.

Key evolution is a commonly used approach for achieving forward secrecy (Ren *et al.*, 2011). In Itkis & Reyzin (2002), the authors have proposed a scheme based on key evolution to combat the problem of key exposure in the context of digital signatures, though it could be applied to other aspects of security as well. They proposed signer-base intrusion-resilient signatures (SiBIR), which incorporate specific data from the

time of generation of the key into the secret key used by each node. Here, the varying key is split into two parts that are separately stored by the signer and the base, and to regenerate the key, both the parts from the signer and base are combined. This scheme preserves security of past and future time periods, when both signer and base are compromised in any arbitrary order, though not simultaneously.

In Ren *et al.* (2013) proposed an optimized secure and reliable data distributed data storage scheme, which takes advantage of both key evolution and RS Codes to ensure data reliability, guarantee forward secrecy, and provide for probabilistic backward secrecy. In this scheme, a sensor node S_i, generates data d_i^r in round r and stores it locally. The scheme uses a single master key K_m and a one-way hash function. The mobile sink preloads each sensor node in the network with the initial data encryption key K_i and hash function $h(.)$. The key K_i is computed as $h(K_m||i)$. At the end of each round, the round index r and the encryption key K_i^r are updated as $K_i^{r+1} = h(K_i^r)$. Forward secrecy is claimed to be maintained, since at each round K_i^r is updated, the adversary cannot derive the key for previous rounds due to one-way property of hash function. The sensed data is encrypted with the round key K_i^r. The encrypted data is then divided into n parts using RS code, and based on node selection scheme the data is sent to neighboring nodes by using pair wise secret key to encrypt the packet, after which the original data is securely erased from the sensor's memory. When the mobile sink visits, it collects all data parts from the nodes and reconstructs the original data. However, if the adversary can compromise the sensor node it can have access to the entire memory of the node, including the algorithm, the round key K_i^r and the round number r. The secret key K_i^r and round r it can still be able to derive the future keys that will be used in the further rounds.

In Di Pietro *et al.* (2013), the authors proposed a solution to data confidentiality and availability issues in UWSN by applying secret sharing and node mobility in the event of node compromise or failure. Also, the authors introduce an energy-efficient scheme based on local secret sharing that uses the mobility of nodes in order to offer information diffusion in WSNs, ensuring confidentiality by setting up secret keys between nodes, which is referred as key agreement problem. Yang *et al.* (2010) addressed the three types of key agreement schemes: trusted server, self-enforcing, and key pre-distribution schemes, and introduced a new key agreement scheme that is hybrid between trusted-server and pre-distribution schemes. The authors claim that schemes which depend on asymmetric cryptography as a self-enforcing scheme are not practical in WSN, due to its complexity and high power consumption. The authors proposed a new architecture of WSN, which gains more shared key and reduce computations. Confidentiality and availability issues of mobile UWSNs were addressed also by Di Pietro *et al.* (2013b), where the authors proposed a new approach leveraging secret sharing and information diffusion, in order to enhance

the confidentiality and availability of data. Availability concern is to minimize data loss due to node failure or capture. Confidentiality is focused to avoid unauthorized access to sensed data. Information sharing between nodes prevent data loss or stealing. Moreover, nodes mobility is used to provide data security using local communication only with configurable parameters that form the trade-off between data confidentiality and availability. Based on simulation and analytical results, the authors concluded that local information sharing is the best fitting solution for efficiently improving security of mobile UWSNs.

In Di Pietro *et al.* (2010), the authors studied intrusion resilience in Mobile UWSNs assuming a powerful adversary for self-healing, and how sensor mobility affect network self-healing. The authors introduced health ration and duty cycle metrics in order to characterize the overall behavior and security of the WSNs. The simulation results and analysis showed that sensor mobility is an effective means for self-healing against mobile adversary, and subsequently proposed another self-healing scheme for UWSN in (Elsafrawey *et al.*, 2014), using cluster controlled mobility (SH-CCM), leveraging mobility within a cluster of sick sensors and hybrid cooperation between sick sensors and other reactive and proactive peers. The authors focused on enhancing the security and reliability of UWSN by increasing the probability of finding healthy neighbors. The proposed algorithm helps the sick sensors to self-heal and restore its backward secrecy using sensor mobility. The results show faster recovery than non-mobile sensors. Three main parameters are used to measure the performance of the proposed approach that are: compromising probability, probability of backward secrecy compromise, and data reliability. The author's analytical and simulation results show that leveraging mobility of sensors improves performance over schemes without controlled mobility.

In Bohli *et al.* (2011), the authors investigated data integrity in UWSN, to which they proposed a new resilient data aggregation scheme leveraging the quality of information (QoI) as protection factor. The authors claimed that QoI metric associated with each aggregation result is necessary for WSN in order to detect attacks, or simply errors. The proposed scheme is able to discover the effect of the attack and also mitigate from it.

In Cheng *et al.* (2014), the authors proposed a hybrid scheme leveraging erasure codes (EC) and self-repairing codes (SRC) in order to provide enhanced security for data storage and redundancy maintenance in WSN. The results show that the hybrid approach provides better performance than EC and SRC schemes. Moreover, the authors proposed to enhance the security of the network by introducing a new location-based repairing scheme, which is claimed to facilitate nodes resistance to mobile adversaries in UWSN.

In Bahi *et al.* (2014), the authors proposed a non-cryptographic scheme to ensure data survivability in UWSN, by providing an epidemic-domain inspired approach in order to model data survivability. The paper focuses on arbitrary dynamic network topologies rather than static networks, as the authors describe it as the novelty of their work. The authors discussed two models susceptible infected recovered (SIR), and susceptible infected susceptible (SIS), which can ensure data survivability assuming an adversary with multiple attack types targeting the network.

NETWORK MODEL

In this section, we define an abstracted WSN for which we develop the proposed approach for data confidentiality. We also discuss the threat model to outline the type of attacks/threats that are being considered in the proposed scheme.

Network model

We consider a UWSN that consists of N stationary sensor nodes that are fixed to specific location in the network coverage area. We denote a sensor node in UWSN as $S_i, (1 \leq i \leq N)$. The network also consists of a mobile sink that visits the nodes in the network at infrequent intervals to collect data from them (Di Pietro *et al.*, 2010). The environment measurement is done periodically with each measuring cycle called a *round*. The sensor node S_i senses data at every round then stores it in its memory and waits until the mobile sink visits to offload the sensed data. Sensor nodes have resource limitations, such as low computational power and small storage memory, whereas the mobile sink is powerful in terms of its communication capabilities, processing power and large storage memory. Each sensor node has the ability to perform symmetric key encryption and one-way hashing.

Threat model

UWSNs, just like any other communication network, can be susceptible to an attack by adversaries. An adversary can compromise the sensor nodes during the period between mobile sink visits, to copy the stored sensed data and the encryption keys used, while trying to avoid detection. An adversary is assumed to be capable of compromising up to $k < N$ sensors, where N is the number of sensor nodes in the network and k is a finite number. An adversary is also assumed to not interfere with the behavior of a compromised sensor or communication between a compromised sensor and other network nodes. It is a read only adversary i.e., would not modify any data sensed by or stored on compromised sensor nodes. The adversary actively compromises or attacks sensor nodes for a finite period of time, reads and copies the storage memory and then listens to all communications of each of the compromised sensors. While sensor nodes can be compromised, the mobile sink, on the other hand, is assumed to be safe, trustworthy and can never be compromised.

PROPOSED SCHEME

Data secrecy

Data secrecy is a fundamental security issue in UWSN. Hence in this section, we present our proposed scheme that aims at providing forward and backward secrecy in UWSN. In the proposed scheme, each time a sensor collects data, it encrypts this data and sends it to one of its neighbors so that when an adversary compromises this sensor, he will not find any data encrypted by this sensor in its memory and hence any key available at the time of compromise cannot be used by the adversary to decrypt any packet in the sensor's memory. When a sensor receives a packet from its neighbors, it uses this packet to update its key. The proposed scheme provides forward and backward secrecy by generating a new key in each round using some data from the packets, which are received from other sensors in the network as we describe the details in this section. With respect to a particular sensor S_i and based on the value of the counter field packets can be classified into:

1. Generated: S_i generates the packet, and fills it with the data sensed by S_i then encrypted by S_i keys.
2. Transient: S_i receives the packet and the counter field is greater than 0. S_i decrements the counter and sends it to one of its neighbors.
3. Resident: S_i receives the packet and the counter field equals 0. S_i keeps it in its memory until the sink's next visit.

Table 1. Summary of Nomenclature

Term	Description
r	The round, the periodic environment measuring cycle
sid	The sensor node ID that forwards the packet to the neighboring node
pid	The sensed data packet ID at round r that is forwarded to the neighboring node
$hops$	The total number of hops the packet, and hence the encrypted data, is to traverse before reaching the residing node
$counter$	The remaining number of hops the packet has to traverse to reach the destination
$KeyGenTable$	All the information used for key evolution is such as PID, SID, B are stored in *KeyGenTable*
w	The mobile sink
k	Number of compromised sensors at any round
S_i	Sensor node *i* in the UWSN
K_i^0	Initial data encryption key
K_i^r	Data encryption key at round *r*
$h(.)$	One-way hash function
b	Specific byte in the data packet used for key evolution
$K_{i,j}^r$	Pair-wise communication key for nodes S_i and S_j at round *r*
$K_{i,w}^r$	Communication key for nodes S_i and the sink *w* at round *r*
N	Number of sensors in the network

The proposed scheme consists of five steps: system initialization, data encryption and packet construction, packet distribution, key generation, and data reconstruction.

Step 1: System initialization

The mobile sink preloads each sensor node S_i in the network with the node's initial data encryption key K_i^0, the number of hops parameter $hops$, and the sensor's hash function $h(.)$. The initial encryption key K_i^0 is used to encrypt the first few packets that are received before the sensor generates a new key. Finally, the number of hops parameter $hops$ determines the number of sensors traversed by a packet before it resides in the memory of the last sensor.

Step 2: Data encryption & packet construction

At each round r, the sensor node senses data D. First, sensed data, D is encrypted

using an asymmetric key encryption algorithm Enc () and a key K (key generation is discussed in Step 4) to produce $encData$

$$encData = E(D, K_i^r)$$

Second, the packet is constructed where it consists of five fields: $pid, sid, round, counter,$ and $encData$ which represents packet id, sensor id, encryption time, number of sensors will be traversed and the encrypted data respectively, as shown in the function *GeneratePKT()* in Algorithm 1. The fields $pid, sid, round$ (during which data encrypted was generated) and $counter$ are sent unencrypted. Figure 1 shows the packet structure.

pid	sid	$round$	$counter$	$encData$

Fig. 1. Packet structure

Step 3: Packet distribution

After packet construction, the sensor sends the generated packet to one of its neighbors. The sensor selects the neighbor that will receive the packet based on a predefined round robin order so that no sensor remains for a long period without receiving any packet. The sink can specify the round robin order for each sensor, such that no two sensors send a packet to the same sensor at the same time and the frequency of receiving packets for each sensor is maximized. When a sensor S_i receives a packet from one of its neighbors, it checks the counter field:

- If $Counter > 0$: S_i decrements $Counter$ by one and forward the packet based on the round robin order specified for S_i to one of its neighbors.
- If $Counter = 0$: S_i stores the received packet in its memory.

This step, shown in the function *ReceivePKT()* in algorithm 1, is performed to ensure that the adversary cannot identify the sensors that store the packets of a specific sensor. In addition, it allows sensor S_i to receive new data that is not known to the adversary who has previously compromised S_i and release it. Therefore, the integrity and forward secrecy of the sensed data is provided.

Step 4: Key generation

Each sensor node generates a new key, when it receives a transient packet during each round r, as shown in function *GenerateRounfKey()* in algorithm 1. When a sensor receives a transient packet, it uses a specific byte of the packet to encrypt the data packet.

$$b = encData_{sid}^{pid}(x)$$

The specific byte (b) field and the previous round key are the inputs to the one-way hash function used to generate other keys at each round.

$$K_i^{r+1} = h\,(K_i^r \,||b)$$

Where, $||$ stands for the concatenation operator, $r = 1, 2, \ldots, K_i^{r+1}$ is the new key generated. The triplet of the location of $b(sid, pid)$ is stored in the sensor's key generation table $KeyGenTable$ for collection by the sink in its next visit.

Step 5: Data reconstruction

When the mobile sink visits a sensor node, the sensor node sends all its resident packets to the sink along with the entire key generation Table $KeyGenTable$. The sink will use the entries in the sensor key generation table to re-generate the entire sensor's round keys to be able to decrypt the sensor's packets.

Communication keys evolution

Above, we only focused on evolving the data encryption keys to protect the data from the compromising adversary, the node-to-node and the node-to-sink communication however is left unprotected. The adversary, as a result of compromising the node, might get the node-to-node pair-wise shared communication keys and the node-to-sink communication keys, and hence can listen to all the communication going out of and coming into the compromised node. Therefore, we propose evolving the communication keys that would help render the adversary blind to data being exchanged between the compromised node and its neighbors and the sink.

The evolution of the peer-wise communication keys can be achieved by supplying the nodes with a 2-input key generating function to generate key $K_{i,j}^{r+1}$, the shared key between nodes *i* and *j* at round *r*.

$$K_{i,j}^{r+1} = h(K_{i,j}^r, r)$$

The Evolution of the node-sink communication keys can also be achieved by supplying the nodes with another 2-input key generating function to generate key $K_{i,w}^{r+1}$, the shared key between nodes *i* and the sink *w* at round *r*.

$$K_{i,w}^{r+1} = h(K_{i,w}^r, r)$$

The result is a compromising adversary that may only be capable to listen to the communication of the compromised node in two cases:

1. When the adversary compromises the victim node and get the communication keys until the communication keys are changed in the next round
2. When the adversary copies the entire memory of the victim node and debug the content to arrive at the original algorithm and key generating function

SECURITY ANALYSIS

In this section, we discuss the robustness of the proposed method in the face of the power of the adversary that want to break this solution and regenerates the keys. When an adversary compromises a sensor S_i, the round encryption key K_i^r and all the information stored in its memory are revealed. Specifically, the adversary can have access to the current key of S_i, all the resident packets, the algorithm and the encrypted byte location triplet.

Forward secrecy analysis

LEMMA 1 (CORRECTNESS). *Forward secrecy is maintained if the max number of hops is set to the network diameter.*

PROOF. The adversary, when compromising a sensor node, has access to the round key and the packets residing in the node. With the number of hops set to the diameter of the network, a sensor's data would be stored at any sensor node in the network. But since the resident packets in the sensor node belong to other sensors, the adversary cannot decrypt these packets with key available in the node. The only case in which the adversary can decrypt the residing packets is when the adversary can have access to all keys in the nodes of the network. But the adversary compromises at most $k < N$ nodes (section Threat model), hence forward secrecy is maintained. □

Algorithm 1 Forward Backward Secrecy

```
1: function RECEIVEPKT()(Packet, j)
2:     if Packet.counter > 0 then
3:         Packet.counter = Packet.counter − 1
4:         GenerateRoundKey(Packet)
5:         Send(Packet, k)                ▷ k is a neighbor picked in predefined round robin
6:     else
7:         Store packet in the node
8:     end if
9: end function
10:
11: function GENERATEROUNDKEY(Packet)
12:     B = Packet[B_location]
13:     K^{round+1}=h(K^{round}, B)
14:     Add(KeyGenTable, Packet.sid, Packet.pid, round)
15: end function
16:
17: function GENERATEPKT(Packet)
18:     encData = Encrypt(Data, K^{round})
19:     Packet = ConstructPkt(i, sequenceNumber, round, hops, encData)
20:     Send(Packet, k)                ▷ k is a neighbor picked in predefined round robin
21: end function
22:
```

Algorithm. 1. Forward backward secrecy

Backward secrecy analysis

LEMMA 2 (CORRECTNESS). *Backward secrecy is maintained if the max number of hops is set to the network diameter and as long as the location not the value of B is stored.*

PROOF. Although the adversary can have access to all information stored in the sensor node memory, including the round encryption key as stated above, the key evolution algorithm render the compromised key useless in the following round. For an adversary to compromise the sensor's information after the adversary had left would require the adversary to be able to regenerate the round key. To be able to regenerate the round key, the adversary needs to have the key evolution algorithm, the current key and the random byte from a random transient packet. Obtaining the byte from a random position in a random transient packet generates the true random process that is more random and hence more secure than a mathematical pseudo random function (key generation explained in step 4 in the previous section). Our scheme does not store the random byte used in the sensor node, but rather store the triplet of its location. The actual byte in a packet is stored in a node somewhere in the network, since the number of hops the packet would traverse is set to the network diameter. Therefore, the adversary would need to re-compromise the node in every round to get the B triplet and then compromise all nodes in the network to find the packet that contains B to successfully regenerate the round data encryption key. But the adversary compromises at most $k < N$ nodes (section Threat model), hence backward secrecy is maintained.

LEMMA 3 (SOUNDNESS). *All Nodes in the network will receive transient packets to generate data encryption keys.*

PROOF. Receiving a transient packet is essential to the successful operation of our proposed algorithm, since it is the way a sensor node evolves its data encryption key. Consider sensor node S_i with g neighbors, all sending their generated packets to their neighbors in a round robin order. Consider the case at round r when the sending sequences of the neighbors of S_i will include node S_i. The maximum number of rounds node S_i not receive a transient packet would be upper bounded by:

$$X = Min\ (NumberOfNeighbors(\ NeighborsOf(S_i)\ )$$

Therefore, no sensor node will stay for more than X rounds without receiving a transient packet for generating a new key.

The sensor node vulnerability period during which this security failure can happen is between consecutive visits of the mobile sink. The more measuring rounds between the mobile sink visits, the more data is at risk in the vulnerability period. Thus increasing the speed of the mobile sink and hence reducing the time between sensor

visits in combination with increasing the time between data measurements can reduce the vulnerability period and as a consequence the risk of data security failure.

EXPERIMENTAL RESULTS

The proposed scheme relies on storage of data packets at each of the sensor nodes in the UWSN. In this section, we evaluate the memory requirements needed to implement the proposed scheme. We simulated the UWSN using a custom simulation program written in C. The simulation models a 250,000 m^2 UWSN with 10000 sensor nodes connected in a random topology. The sensor nodes are static with a coverage area of radius 10 m. The nodes make their measurements and generate data, round times, every 10 minutes. The sink roams about the network to collect the data from the sensor nodes with a speed of 4 m/s. The data packet is designed to be of 5 fields as described in step 2 of the proposed method with a size of 2 bytes for each filed for a total packet size of 10 bytes. We investigated the impact of the number nodes in the network and the number of hops on the memory requirement in the sensor node. We run the simulation program for several network sizes and number of hops that ranged from 5 hops to a number equal to the diameter of the network, assuming the network was square shaped. In all these simulation scenarios, we were interested in the maximum memory required by the sensor nodes, since this value represents the maximum data stored in a node until it offloads the stored packets to the sink, when it visits.

Figure 2 shows that the additional memory required by the proposed method does not burden the limited memory available to the nodes and is well within the available memory of current commercial motes. The figure also shows how the proposed scheme does not utilize more of the memory as the number of nodes in the network increases. This is due to the fact that the proposed scheme only affects nodes number of hops away from the reference node, so as the number of nodes increases in the network, far away nodes are not affected.

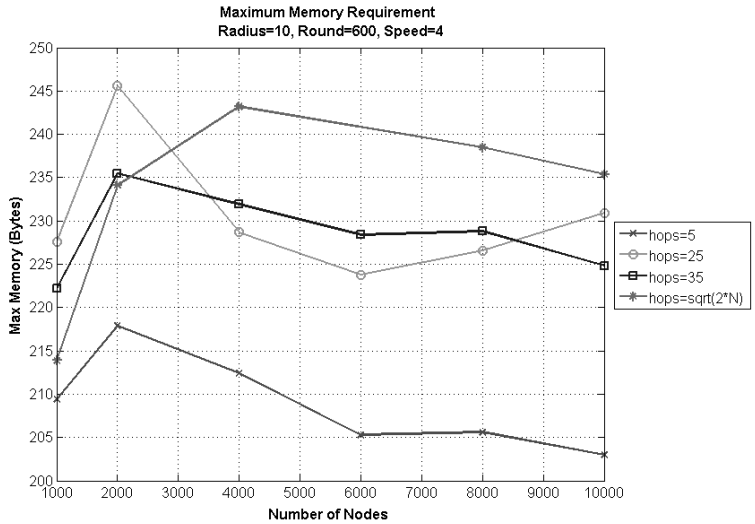

Fig. 2. Memory requirements with varying number of nodes and hops

Figure 2 also shows the impact of increasing the number of hops until its residing node on the memory requirement. Although the simulation result shows that, as we increase the number of hops more memory is required, the additional requirement (a maximum of about 35 additional bytes) is not significant compared to the memory available to current commercial motes. This increase in memory requirement can be explained by the fact, as the number of hops increase, nodes would receive packets from more possible nodes. Since we are interested in the maximum memory requirement, with more possible packet senders to a sensor node, the greater the probability of receiving more packets, and hence greater number of received residing packets. The effect of the sink speed on the memory used in the sensor is also studied. Sink inter-visits time is inversely proportional to the sink speed, and the longer that inter-visit period is, the more the sensor will have data packets residing in it and hence more memory is required and used. This is reflected in the results of Figure 3, where the maximum memory used by the proposed scheme is decreasing, when we increased the sink speed from 4 m/s to 8 m/s and 12 m/s. As the speed of the sink increases, the inter-visit time decreases and packets are offloaded to the sink in less time, which leads to less number of packets being allowed to accumulate in the sensor node, and hence less sensor node memory is required by the proposed method.

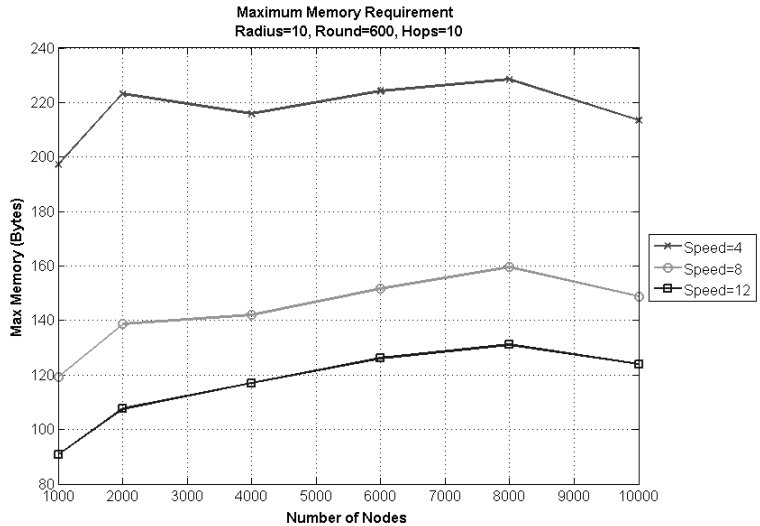

Fig. 3. Memory requirements with varying value of sink speed

Let node neighbor density be the average number of neighbors per node in the network. Figures 4 and 5 illustrate the affects of node density and number of hops in the proposed scheme on node memory usage. The figure shows that as the node density degree is increased, the used node memory is decreased. This is due to the node cycling through a larger set of neighbors to forward packets to which in turn would involve more nodes in the network in the packet forwarding process, hence spreading the burden of storing residents, packets among a larger set of network nodes.

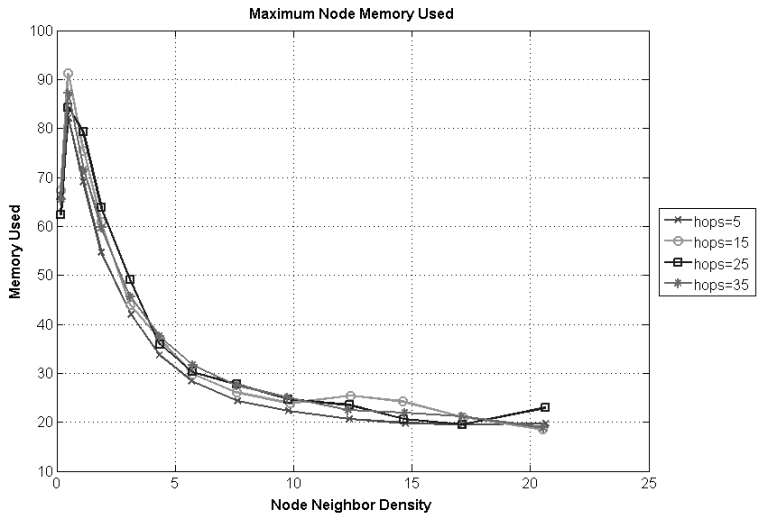

Fig. 4. Maximum node memory used as a function of node density

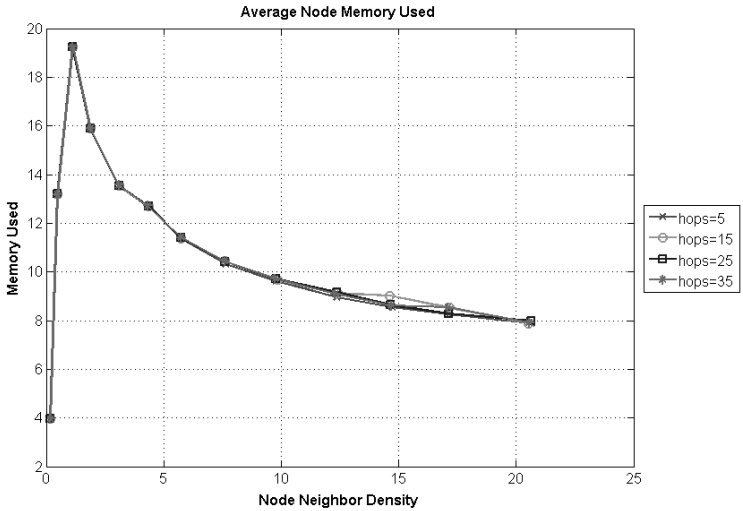

Fig. 5. Average node memory used as a function of node density

It is of interest to investigate the dynamics that affect the number of transient packets received by any one node, since transient packets are used to generate the key used to encrypt the data sensed during that round (see step 4 of the proposed scheme). It is vital to the robustness of the key generation function that the node receives a transient packet to ensure the freshness of the round key. Therefore, we ran experiments to study the minimum number of transient packets received by a node in the network. Figure 6 shows the effects of increasing the density and the number of hops on the minimum number of transient packets arriving at a node. The figure shows that as the number of hops is increased, the number of transient packets per node is also increased. This is due to the fact that, when the value of hops is increased, a packet has to traverse more nodes increasing the average number of packets arriving at a particular node. On the other hand, the figure shows that as the density of nodes increase, the minimum number of transient packet per node is in fact decreasing. This is due to the fact that a node forwards a transient packet to only one of its neighbors, and as the number of neighbors increase, transient packets are forwarded to a larger set of neighbors, reducing the minimum number received by any one neighbor. These effects on the minimum number of transient packets passing through a node are of interest, since transient packets are essential to round key generations and their number should be kept non-zero.

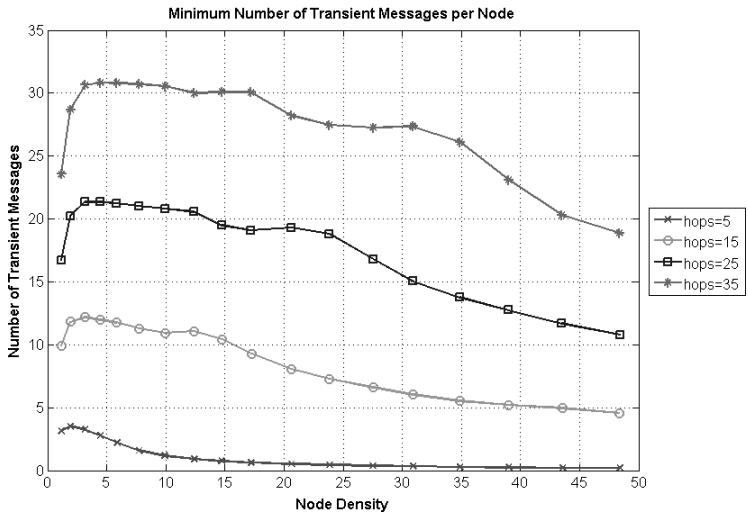

Fig. 6. Minimum Number of transient packets as a function of node density

Figure 7 illustrates the effects of number of hops and node density on the number of transfers a node would need to perform. The figure shows that the number of transfers a node performs is directly proportional to the number of hops, but indifferent to the degree of node density. This is due to the fact that the maximum value of hops is essentially the number of times a packet is passed to a neighbor, thus the greater the number of hops the greater number of transfers performed by the network nodes. Node density on the other hand does not introduce any additional packet transfers since a node transfers a packet only once and to one neighbor only, regardless of the number of neighbors it has.

Regarding the energy consumption of our proposed algorithm, it is obvious that forward and backward secrecy relies on the number of messages sent by a node to its neighbors. However, there is a tradeoff between the robustness of the proposed scheme and the energy consumed as a result of it. The greater the number of maximum hopes set in the algorithm, the more robust the security of the data generated will become (see Lemma 1). However, a greater maximum number of hops value would lead to a greater number of packet transfers to be performed by a network node consuming more energy.

The computation complexity of our algorithm due to encryption is low, since messages are only encrypted using a one-way hash function. Decryption is performed only by the sink node, which is assumed to have unlimited power. The complexity of generating new keys is also done by low complex functions.

In each round, a node performs one packet generation and one key generation actions, and forwarding received transient packets to neighbors. Let d be the number

of nodes traversed by a generated packet until it reaches the residency node. Therefore, in each round, n packets are generated in the network, reaching a steady state number of transient packets of n*d after d rounds. Therefore, in each round, a node would perform one packet generation action, one key generation action, and on average forward d packets. Hence, it can be clearly seen that the computational complexity of the proposed scheme is O(d) actions per round. This finding is supported by the results shown in Figure 7, where as the number of hops, d, increases the number of communications with neighbors increases.

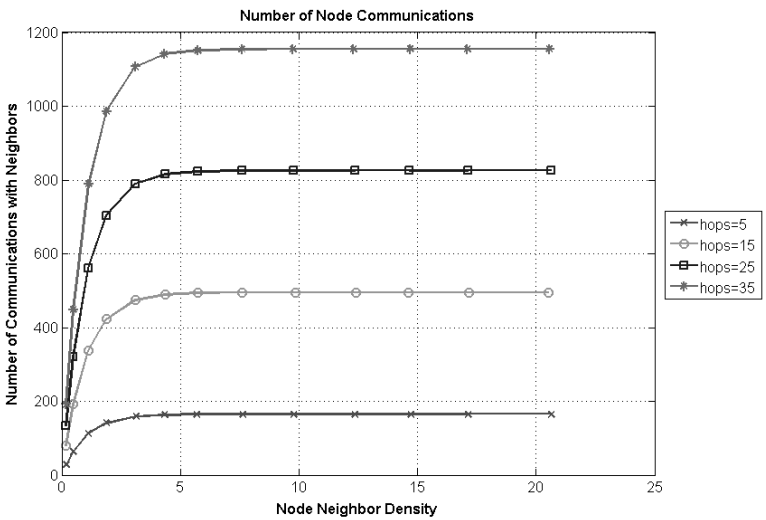

Fig. 7. Average node communication as a function of node density

FUTURE WORK

In this section we want to present some future extensions to the proposed algorithm in order to increase the security and introduce energy-efficiency to the sensor network. In subsection network model, we assumed that sensors belonging to the network are usually trusted. Therefore, we did not consider the case of foreign sensor nodes joining the network. Since in this case, there is a risk of fraud nodes, which could be integrated into the network to spy out the communication, we have to introduce a trust mechanism to our approach. To build trust between two or more nodes, a shared secret is needed, which can be based on security keys. Only nodes that know the secret will be able to participate in the network. Since, the nodes are stationary, we will assume that a node joins the network only once, i.e. rendezvous situations are excluded. The integration process of a new node should be managed by a trustworthy node. Therefore, we propose to use the mobile node to activate a new node in the network. For securing the communication, mobile and applicant node should be locally close together and should use short communication range. During this critical

phase, all security keys should be generated. Which keys and how they are initiated, will be part of our future work. The trust mechanism can also be extended to a life-time trust matrix managed by each node to supervise trust level of neighboring nodes. Furthermore, we plan to introduce passive participation of nodes to our approach, i.e. nodes listen passively to communication of their neighbors. If a node discovers an anomaly in behavior of a neighbor, like the neighbor does not forward the message as it should do, the actual node takes additional countermeasures. This could be sending the message additionally to other neighbors.

Another extension of our approach is to introduce hierarchical networks. Up to now, our network model allows a flat structure, where each node has the same tasks and responsibilities. This is an easy to manage network, but does not scale if number of nodes increase drastically. Having a hierarchical network brings additional advantages, if there are also energy requirements. This was neglected in our proposed scheme, assuming each node has its stable and unlimited energy source. If nodes work on battery power, energy-efficiency becomes highly important. In future work, we want to examine cluster-based networking models for our proposed approach.

An alternative method to save resources is to dynamically adopt the hop range of reference node. For now, the hop count for sending the sensor data is fixed to the same value for all nodes in the network. As shown in previous section of experimental results, the number of hops influences traffic intensity and memory usage. Therefore, it would be interesting to examine an approach, where hop count is dynamically set by each node based on traffic and resource constraints. In terms of battery operated nodes, this feature would provide more flexibility in energy management.

As more neighbors a node has, more it will need to communicate, leading to higher energy consumption. In future work, we want to propose an extension of our algorithm, where we will pass remaining energy information of nodes to its neighbors. This additional information can be used to decide, if a node will send a message to its neighbor or not. If a neighbor has remaining energy below a defined threshold, the current node will skip this neighbor in the next communication round. This dynamic scheme will bring energy balancing features into our proposed algorithm.

CONCLUSION

In UWSNs, data is susceptible to the risk of being revealed, due to the absence of an online-sink, which makes data confidentiality a major concern and challenge. In this paper, we have proposed a distributed scheme to ensure data confidentiality in UWSN that provides forward and backward secrecy in the presence of a mobile adversary. Forward secrecy and backward secrecy are provided through key evolution and data distribution scheme. Through experimental results, we have shown that the memory overheads of such a scheme are minimal.

ACKNOWLEDGEMENTS

This research was supported by the Kuwait University Research Administration under grant EO02/12.

REFERENCES

Bahi, J.M., Guyeux, C., Hakem, M. & Makhoul, A. 2014. Epidemiological approach for data survivability in unattended wireless sensor networks. In Journal of Network and Computer Applications 46 (2014): pp. 374-383.

Bohli, J.M., Papadimitratos, P., Verardi, D. & Westhoff, D. 2011. Resilient data aggregation for unattended WSNs. In Local Computer Networks (LCN), 2011 IEEE 36th Conference on, pp. 994-1002. IEEE, 2011.

Cheng, W., Li, Y., Jiang, Y. & Yin, X. 2014. A novel secure and repairable scheme for distributed data storage in unattended WSNs. In Industrial Electronics and Applications (ICIEA), 2014 IEEE 9th Conference on, pp. 2154-2159. IEEE, 2014.

Di Pietro, R. & Guarino, S. 2013. Data confidentiality and availability via secret sharing and node mobility in UWSN. In INFOCOM, 2013 Proceedings IEEE, pp. 205-209. IEEE, 2013.

Di Pietro, R. & Guarino, S. 2013. Confidentiality and availability issues in mobile unattended wireless sensor networks. In World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a, pp. 1-6. IEEE, 2013.

Di Pietro, R., Ma, D., Soriente, C. & Tsudik, G. 2012. Self-Healing in Unattended Wireless Sensor Network, In ACM Transactions on Sensor Networks, vol. 9, no. 4, pp. 39: 1-19.

Di Pietro, R., Ma, D., Soriente, C. & Tsudik, G. 2008. POSH: Proactive co-operative self-healing in unattended sensor networks, in Proceedings of the IEEE International Symposium on Reliable Distributed Systems (SRDS ,08), Napoli, Italy, 2008, pp. 185-190.

Di Pietro, R., Mancini, L., Soriente, C., Spognardi, A. & Tsudik, G. 2008. Catch me (if you can): data survival in unattended sensor networks, in Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom '08), Hong Kong, China, 2008, pp. 185-194.

Di Pietro, R., Oligeri, G., Soriente, C. & Tsudik, G. 2010. Securing mobile unattended WSNs against a mobile adversary. In Reliable Distributed Systems, 2010 29th IEEE Symposium on, pp. 11-20. IEEE, 2010.

Elsafrawey, A.S., Hassan, E.S. & Dessouky, M. 2014. Improving UWSNs security and data reliability using a cluster controlled mobility scheme. In Informatics and Systems (INFOS), 2014 9th International Conference on, pp. PDC-21. IEEE, 2014.

Itkis, G. & Reyzin, L. 2002. SiBIR: Signer-Base Intrusion-Resilient Signatures. In Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology (CRYPTO '02), Springer-Verlag, London, UK, UK, 499-514.

Ma, D. & Tsudik, G. 2008. DISH: distributed self-healing, in Proceedings of the International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS'08), Detroit, MI, USA, 2008, pp. 47-62.

Ren, Y. Oleshchuck, V. & Li, F.Y. 2010. A scheme for secure and reliable distributed data storage in unattended WSNs, in Proceedings of the IEEE Global Telecommunications Conference, Miami, USA, 2010, pp. 1-6.

Ren, Y., Oleshchuk, V. & Li, F.Y. 2009. Secure and efficient data storage in unattended wireless sensor networks, in Proceedings of the Third International Conference on New Technologies, Mobility and

Security (NTMS '09), Cairo, Egypt, 2009, pp. 1-5.

Ren, Y., Oleshchuk, V., Li, F.Y. & Ge, X. 2011. Security in Mobile Wireless Sensor Networks - A Survey, In Journal of Communications, Vol. 6, No. 2, April 2011, pp. 128-142.

Ren, Y., Oleshchuk, V. & Li, F.Y. 2013. Optimized secure and reliable distributed data storage scheme and performance evaluation in unattended WSNs, In Computer Communications, Volume 36, Issue 9, 15 May 2013, pp. 1067-1077.

Sen, J. 2009. A Survey on Wireless Sensor Network Security, in International Journal of Communication Networks and Information Security (IJCNIS), Vol. 1, No. 2, August 2009, pp. 55-78

Yang, S., Liu, J., Fan, C., Zhang, X. & Zou, J. 2010. A new design of security wireless sensor network using efficient key management scheme. In 2nd IEEE InternationalConference on Network Infrastructure and Digital Content (2010) pp. 504-508, IEEE, 2010

Yick, J., Mukherjee, B. & Ghosal, D. 2008. Wireless sensor network survey, Computer Networks, Volume 52, Issue 12, 22 August 2008, Pages 2292-2330

Submitted: 15/11/2015
Revised: 16/05/2016
Accepted: 22/05/2016